**Conclusion:**

Graphical user interface, text, application

Description automatically generated

**Value Iteration: (Finding optimal value function + 1 policy extraction)**

* Each iteration updates both utilities (explicitly, based on current utilities) and the policy (possibly implicitly, based on current utilities)
* In **value iteration**, you start with a random value function and then find a new (improved) value function in an iterative process, until reaching the optimal value function. Notice that you can derive easily the optimal policy from the optimal value function. This process is based on the *optimality Bellman operator*.
  + ***The optimality Bellman operator contains a max operator, which is non linear and, therefore, it has different features.***
* Value Iteration works on principle of “ Optimal value function —-> optimal policy”.

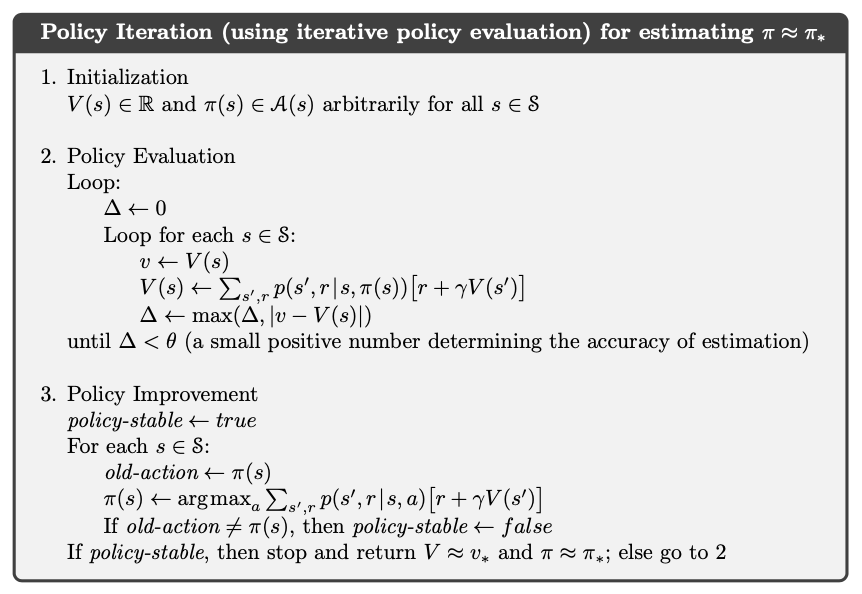
**Policy Iteration: (PE and PI repeats iteratively until policy converges)**

* Policy Evaluation with K value:
  + Does several iterations to update utilities for a fixed policy
  + Repeat until value converges.
    - If we set K = 1, PE will not iteratively call on Bellman update → no values converge. This will then be value iteration instead.
* Policy Improvement:
  + Occasional iterations to update policies
* In **policy iteration** algorithms, you start with a random policy, then find the value function of that policy (policy evaluation step), then find a new (improved) policy based on the previous value function, and so on. In this process, each policy is guaranteed to be a strict improvement over the previous one (unless it is already optimal). Given a policy, its value function can be obtained using the *Bellman operator*.
* Policy iteration works on principle of “Policy evaluation —-> Policy improvement”.

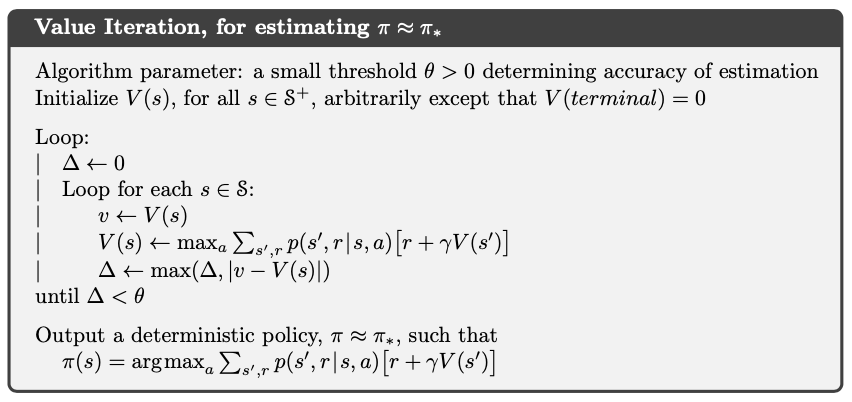
Both value iteration (VI) and policy iteration (PI) algorithms are guaranteed to converge to the optimal policy, so it is expected that you get similar policies from both algorithms (if they have converged).

However, they do this differently. VI can be seen as truncated version of PI.

Let me first illustrate the pseudocode of both algorithms (taken from [Barto and Sutton's book](http://incompleteideas.net/book/RLbook2020.pdf)), which I suggest you get familiar with (but you are probably already familiar with them if you implemented both algorithms).

[](https://i.stack.imgur.com/kKZx7.png)

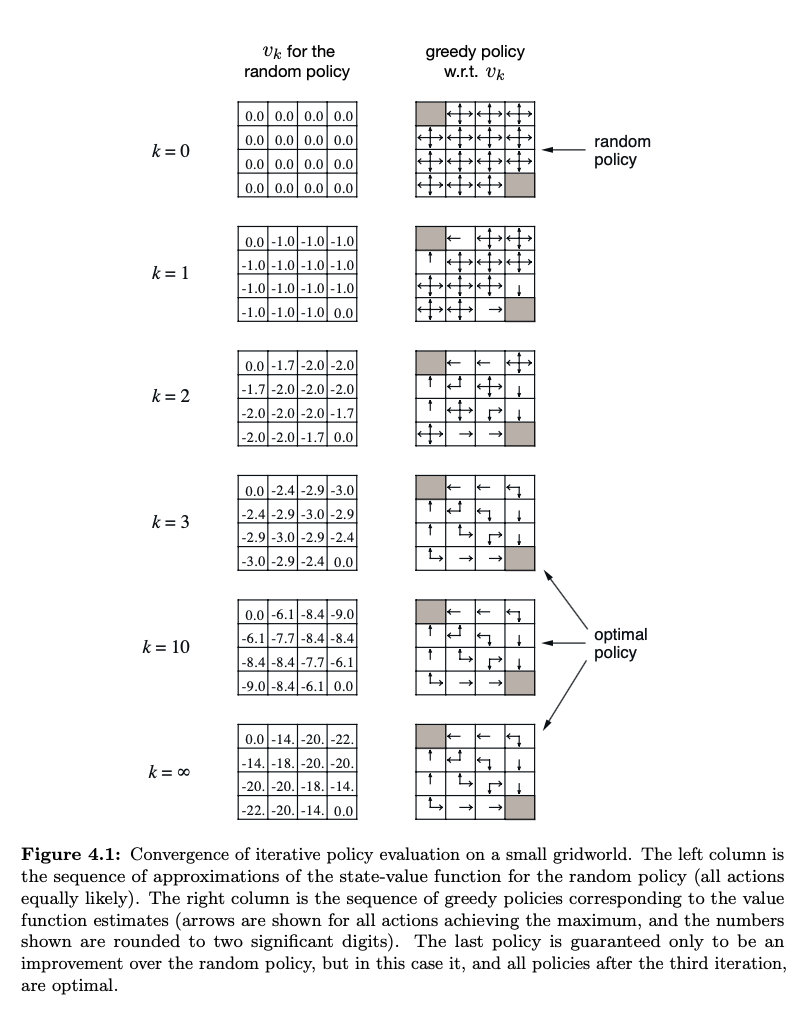
As you can see, policy iteration updates the policy multiple times, because it alternates a step of policy evaluation and a step of policy improvement, where a better policy is derived from the current best estimate of the value function.

[](https://i.stack.imgur.com/CAAu5.png)

On the other hand, value iteration updates the policy only once (at the end).

In both cases, the policies are derived from the value functions in the same way. So, if you obtain similar policies, you may think that they are necessarily derived from similar *final* value functions. However, in general, this may not the case, and this is actually the motivation for the existence of value iteration, i.e. you may derive an optimal policy from an non-optimal value function.

Barto and Sutton's book provide an example. See figure 4.1 on page 77 ([p. 99 of the pdf](http://incompleteideas.net/book/RLbook2020.pdf#page=99)). For completeness, here's a screenshot of the figure.

[](https://i.stack.imgur.com/DwyQL.png)